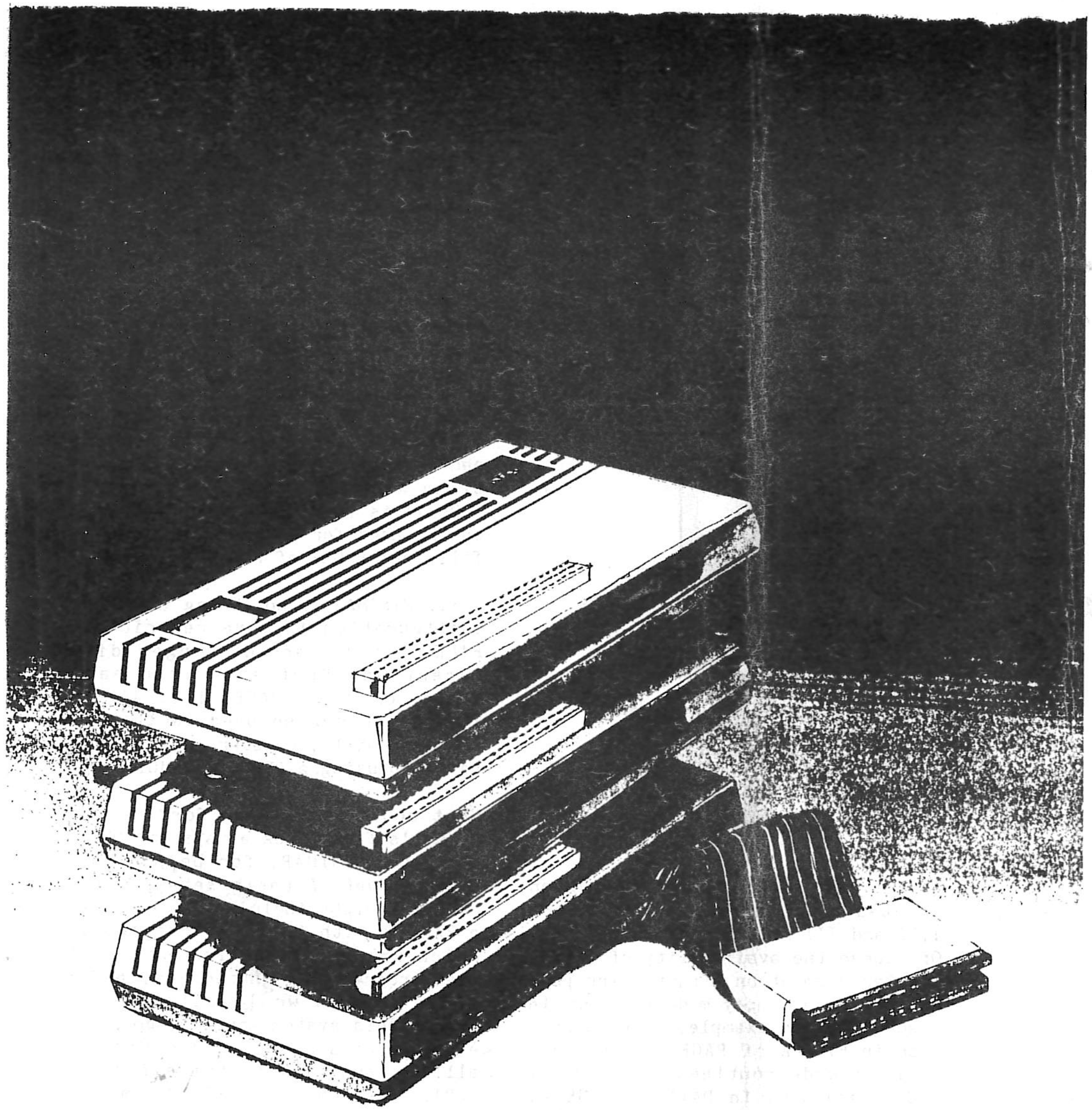


HOW TO GROW UP TO THE MEGABYTE MICRO

Organic Micro for the ZX81

An Introduction to Its Organisation and Applications

Background to the Megabyte ZX81

The concept of packaging a whole microcomputer system in one box would be fine if the answers to the questions: "What problems do I want to solve?" and "How do I want to solve them?" were known. The difficulty is that the first question can be very diverse and the best solution to the second is heavily dependent on existing hardware and software.

Peter Choi of BASICare has solved the problem of satisfying these criteria by producing a modular system which can be expanded from a hobbyist's micro, through to a professional's business system, simply by the addition of extra building blocks.

The limitations of a conventional 8-bit microcomputer soon become evident to anyone who has used one: the memory size is limited to 64K, and expansion of the system is limited to a few ad hoc add-ons. Using BASICare Organic Micro modules you can add an unprecedented amount of extra memory to the ZX81 and choose from a large and expanding range of additional modules to suit any application.

What does it look like?

Organic Micro comes in a family of identical modular units, carefully designed to be both stylish and functional. The size of each module is a compact 186x86x20 mm. Each carries its own physical and electrical connections so that units can be connected into a linked multifunctional assembly which can simply be modified or extended.

In this way a microcomputer can grow from a simple starter system into a fully fledged machine as your needs evolve. The system will always be smartly packaged at each stage of its expansion because of the way the identically packaged modules interlock.

How is it done?

The module at the heart of the system, and on which it grows, is called the **PERSONA**. Its presence is essential for the other modules' use. It buffers in all the signals from the ZX81 expansion port via a ribbon cable. It then talks to other modules via a 64-way ORGANIC BUS configured so that it fits onto the given ZX81 Memory Address Map. Up to 4 Mbytes of addressable positions are allowed, all signals for dynamic memory operation are generated on board. Other modules respond when PERSONA sends their individual SLOT signal along the bus. A generous +/-5 volt regulated DC supply powers additional modules.

MINIMAP is the key module for adding large amounts of memory to the ZX81, though a few memory modules can be used without it. It extends the 64K of the ZX81 address space up to a possible 1 Mbyte. The space is organised into vertical 64K PAGES divided into SEGMENTS labelled: FILE A, FILE B, ROM, TOOL, DATA, PATH, and SLOT. Up to 16 horizontal PAGES can be supported. All this memory can be defined dynamically from within a program, allowing, for example, a program in a FILE SEGMENT to manipulate the contents of DATA SEGMENTS.

Of the other modules several consist of various forms of memory expansion, some are peripheral devices, and some are for communication with the outside world.

RAM 08 is a low-cost basic memory module which has 2K of memory expandable up to 8K simply by plugging in extra RAM chips in the vacant sockets. It is located in a region of address space not used by BASIC programs and is ideal for data or machine-coded routines which might be shared by several programs. The remarkably low cost of the **RAM 16** memory expansion module demonstrates the economy achieved by generating all signals necessary to operate the dynamic memories from PERSONA. On-board address decoding permits the simultaneous use of many RAM 16 modules in conjunction with MINIMAP. In common with all the memory modules it can be enabled/-disabled under the control of an external signal. **RAM 64** provides 64K of memory expansion arranged as four blocks of 16K all of which can be used simultaneously (unlike other available 64K add-ons) under the control of MINIMAP.

The sophisticated and useful **DROM** module is a non-volatile memory of 2K expandable to 8K. This is achieved using ultra low power CMOS static memory in conjunction with a Nickel-Cadmium battery, which is continuously recharged during normal operation. An invaluable module, especially for saving programs under development or often-used routines or data, it makes tedious cassette saving and reloading unnecessary. Individual 2K blocks can be protected against accidental overwriting.

Up to four 2K EPROMS can be used in **TOOLKIT**, which is located in address space after the ZX81 on-board ROM. This allows the calling of TOOLKIT-held routines using the USR function. Since they can be individually enabled/disabled, several TOOLKIT modules can be used under MINIMAP control. **USERFONT** is an option which allows the ZX81 user to define a set of 128 display characters dynamically under program control, when used with DROM or RAM 08. It is also available on TOOLKIT where the 128 alternative display characters will be stored in an EPROM.

A class of modules collectively called **PERICON** provide very general input/output capabilities. **PERICON-a** is a general purpose peripheral interface providing 24 lines of Input/Output. Up to four such modules can be simultaneously used allowing basic I/O, strobed I/O, and bidirectional 8 bit data transfer with handshakes. **PERICON-b** is for access and control of the outside world. The 24 lines are capable of directly operating relays or driving long signal lines. On-board address decoding allows the simultaneous use of up to four such modules. **PERICON-c** provides a centronics parallel interface permitting the ZX81 user to drive a 80-column dot-matrix printer.

How is the memory organised?

The memory is arranged in 64K **PAGES** where each page has the structure of figure 1. The starting address of each **SEGMENT** has been included.

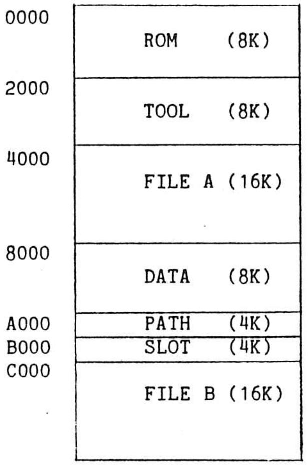


figure 1

The organisation has been designed to fit in with the ZX81 and will be different for another micro, when the relevant PERSONA is available.

Each SEGMENT in the 64K PAGE has a specialised use ultimately dictated by the design of the ZX81. The ROM area is the ZX81 ROM, and contains the BASIC interpreter and operating system.

PATH is for future applications and will not be described here, SLOT is used in connection with the peripheral modules.

FILE A is the normal residence of BASIC programs. FILE B is used by the display but can be used for BASIC if no display is used. TOOL is the best place for your own machine coded routines, and can also be used for data. It can contain your own character set description created with the USERFONT option. DATA is extra space for data, and can also be useful for very long BASIC programs.

With MINIMAP in use very large amounts of memory space is available. It is then possible to have many BASIC programs, and several TOOL and DATA SEGMENTS distributed between different PAGES. It is quite important to understand the concept of PAGE as applied to this system. In figure 2 the different SEGMENTS have been regrouped for convenience of explanation.

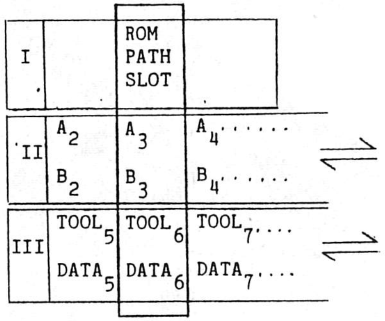


figure 2

A PAGE consists of any 64K column created by sliding memory groups I,II and III relative to each other. Of course the availability of a particular segment on a particular page depends on a memory module being in place. So, for example, a BASIC program in FILE A of PAGE 3 could use machine code routines in TOOL of PAGE 2 and data in DATA of PAGE 2. It would also have access to ROM since all PAGES contain the ZX81 ROM. Note that a program cannot use TOOL or DATA from another PAGE at the same time. However, since the construction of PAGES by sliding the three groups of SEGMENTS can be controlled dynamically from within a BASIC program, using a simple POKE instruction, other TOOL and DATA SEGMENTS could be used at a later time. Notice also that many BASIC programs can be simultaneously resident in different PAGES. It is possible to instantly switch between these different programs. One might envisage, for example, using the output of a program in one PAGE as the input to a general graphics program resident in another PAGE. MINIMAP will support sixteen PAGES of TOOL and DATA.

The position in which a given memory module sits in the memory map can either be set by patching connectors on the back of the modules or they will reside in default positions, in which they are usually most useful. Modules RAM 08 and DROM can be used in TOOL and DATA positions and the natural place for TOOLKIT is in a TOOL SEGMENT. RAM 16 can be used for FILE A, RAM 64 will fill four PAGES of FILES A or B.

If MINIMAP is not being used the configuration is more restricted. FILES A and B are no longer different and "reflect" into each other. Only one PAGE, i.e. 16K of RAM 64 can now be used so this is not worthwhile, however, RAM 16 can be PAGED using a PERICON module.

What could it be used for?

Let's first look at systems which don't use MINIMAP, for people on a small budget. A basic set-up for a hobbyist might include, for example, a RAM 08 with USERFONT option, in which was 2K of RAM for use with user programmable characters, and a RAM 16 which would sit in FILE space. This system would be good for e.g. graphics games, and could use all the usual 16K software for the ZX81. Addition of a PERICON-b module would allow e.g. the control of mains operated devices through relays, or the use of moving LED displays of the shop window type. For Input/Output experiments use PERICON-a, e.g for piano keyboards in use with the projected sound synthesiser module. PERICON-c would allow high quality printing and would be useful for a small shop or business for e.g. label printing, word processing and for printing circulars.

If you bought a DROM you would find it very useful for developing programs and for saving up copies of machine code routines that you might write yourself or copy from journals. Such utilities as renumber or left-right scrolling for example. For the software writer the combination of RAM 16, DROM and PERICON-c would be perfect.

Later on, as they become available, the hobbyist would want to add a joystick controller and sound synthesiser to add to games excitement. An RS232 serial data link might also be needed.

For machine control applications up to four of each PERICON module can be used together to give a total of 96 high current output lines, plus 96 I/O lines and four CENTRONICS printer drivers. A-D and D-A modules will be available in the future.

Larger systems can be constructed using MINIMAP. Up to 16 BASIC programs could be co-resident sharing the resources of up to sixteen DATA and TOOL SEGMENTS. This is usually called multi-tasking and is normally only available on very expensive systems. Some of these programs could be utilities such as graphics, compilers or an editor.

By using data files in FILE A and FILE B space, manipulated by TOOL-held machine code routines an enormous fast access data-base can be created. A menu-driven enquiry system using this would be very useful for a club or business. The fast cassette machine we are developing will get round the problem of loading and saving such large amounts of data. This system compares very favourably with floppy disc drives in terms of price and performance, and allows genuine random access to large amounts of data.

For animated graphics games the system is unrivalled, allowing, for example, the multiplexing of complex backgrounds with moving foregrounds on several screens, each possibly controlled by a different program.

Projects currently in the pipeline include highres colour graphics and colour modulator, very similar to that on the Sinclair Spectrum; and the replacement of the ZX81 by a 16-bit processor, producing the first true 16-bit home computer.

MODULES	RECOMMENDED PRICE £
PERSONA	24.95
MINIMAP	29.95
RAM 08 (2K version)	19.95
RAM 16	21.95
RAM 64	64.95
DROM (2K version)	32.95
TOOLKIT	17.95
PERICON-a	22.95
PERICON-b	27.95
PERICON-c	34.95
OPTIONS:	
USERFONT	6.95
Additional 2K for RAM 08	5.50
Additional 2K for DROM	6.50

Mail order customers add £1.50 postage and packing.
Please add VAT 15% to all prices.

BASICare MICROSYSTEM LIMITED
5 DRYDEN COURT, LONDON SE11 4NH TEL: 01-735 6408
TELEX: 8813271 GECOMS G

Technical Sheet 1

PERSONA

Operational Description

1) The unit connects to the ZX81 via the ribbon cable. It buffers in these address, data and control signals.
2) Address decoding - the module examines the high address bits which map into some of the different 4K and 8K SEGMENTS of a PAGE and produces activating signals for the memory modules sitting in these memory locations. The ROM, TOOL, DATA, and SLOTS 0-7 SEGMENTS are decoded by PERSONA. Peripheral modules such as MINIMAP and PERICON are activated by signals appearing on SLOT pins on the ORGANIC BUS. Patching leads from rear pin-wafer connections allows any 8K module to sit in any 8K SEGMENT. This allows great flexibility in configuring a system, however, the standard set-up requires no patch leads.
3) Dynamic RAM refresh signals are produced.
4) Power supply - a stabilised -5v. supply at 20mA; and an additional +5v., 1A regulated supply are provided. PERSONA draws power from the ZX81 5v. supply. The on-board 5v. regulator powers all modules on the ORGANIC BUS. A 12v. nominal supply is obtained directly from the ZX81 power supply via the ribbon connector.

Rear pin-wafer connections

Power group (left hand group viewed from rear)

1	2	3	4	5	6	7	8	9
REG	+5	+5	+12	+12	0	0		-12

1) Use on-board regulator by shorting pins 1 & 2 with supplied shunt (normal setup). If a large number (>5) of modules are used an external power supply can be added between +12v. and ground (4 & 6). This need not be regulated. Alternatively, for a large, fully developed system a replacement power supply giving a regulated +/-12v. and +5v. can be connected, replacing the ZX81 power supply.
2) Connection of a 12v. battery between pins 5 and 7 via a special connector available from BASICare will protect against temporary loss of power supply. Incorrect connection would be very dangerous.

Module location group (right hand group viewed from rear)

1	2	3	4	5	6	7	8	9
ZXRAM DISABLE		RESET		TOOL	DATA		ROM	ZXROM

1) Shunt 1 & 2 to disable ZX81 RAM (usual setup).
2) Shunt 3 & 4 to reset the current PAGE, and to initialise as for normal ZX81 power-on, leaving FILE A and FILE B in other PAGES and all TOOL and DATA contents intact.
3) Designate up to 8K of RAM as sitting in TOOL space by patching a lead from pin 5 to memory module.
4) A patch to pin 6 designates memory as sitting in a DATA SEGMENT of the memory map.
5) Pin 8 is similarly for designating memory modules to the ROM SEGMENT.
6) Pin 9 is ZXROM disable. Disconnect from 8 for some games e.g Raiders, which use the fact that the 8K of a PAGE after the ZXROM is a copy of the ZXROM.

Technical Sheet 3

RAM 16 and RAM 64

Operational Description

RAM 16 is a 16K block of random access memory, RAM 64 has four such 16K blocks in one module. The four 16K blocks of RAM 64 will each sit in a different PAGE of memory (see "Organic Micro for the ZX81", fig. 2). Note that without the MINIMAP module only one 16K block of RAM 64 can be used. These 16K blocks of memory sit in section II of the memory map, (see "Organic Micro for the ZX81", figure 2) either in the FILE A or FILE B SEGMENTS.

It is important to realise that FILE B of a particular PAGE can only be used as separate from FILE A of the same PAGE when it is being used for storage of programs or data in a PAGE in which you do not wish to make use of the display. This is because the ZX81 expects to find a copy of the display file, which originates from FILE A in this area of memory and will crash, of course, if it doesn't.

Rear pin-wafer connections

rear view

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
A/B		PAGE 0		PAGE 1		PAGE 2		PAGE 3		BANK 0		BANK 1		BANK 2		BANK 3	

In the megabyte memory address space there are 16 PAGES of FILE A and 16 PAGES of FILE B, each FILE being 16K long. The 16 PAGES are arranged as four BANKS (BANK 0,1,2,3) with four PAGES in each BANK (PAGES 0,1,2,3). For example the fourth PAGE is labelled as BANK 0, PAGE 3, and the fifth is BANK 1, PAGE 0, etc.

You must tell each 16K block which PAGE and which BANK it is in, and whether they are all in FILE A, FILE B or both.
1) Shunt pins 1 & 2 so that each 16K block sits in both FILE A and FILE B of each of PAGES 0-3 if you wish to use the display. This is the mode in which an ordinary Sinclair 16K Rampack operates.
2) You can make all the 16K blocks sit in either FILE A or FILE B of their PAGES by connecting a special twin-wire jumper from pins 1 & 2 to one of the MINIMAP FILE A or FILE B pins.
3) Position the module in one of BANKS 0-3 by shunting one of 11&12 (BANK 0); 13&14 (BANK 1); 15&16 (BANK 2) or 17&18 (BANK 3).
4) Select a RAM 16's PAGE number by shunting one of: pins 3&4 (PAGE 0); pins 5&6, (PAGE 1); pins 7&8, (PAGE 2); pins 9&10, (PAGE 3). The owner of a RAM 64 will naturally want to enable all four PAGES within the selected BANK. This is done by shunting pins 3&4, 5&6, 7&8 and 9&10.

IMPORTANT - Do not allow the BANK designations on different RAM 64 modules to conflict since if you try and read from two memory modules with different contents sitting in the same address they will probably be damaged.

NOTE - See MINIMAP Technical Sheet 2 for advice on how to manipulate PAGES.

Technical Sheet 2

MINIMAP

Operational Description

1) MINIMAP is a peripheral, and sits in the SLOT 0 area of memory.

The unit generates two sets of four extra address bits which are the page addresses of SECTIONS II and III in the WORKPAGE (see fig. 2 of "Organic Micro for the ZX81"). This is simply controlled by POKEing a number into SLOT 0. The four extra address bits give 16 times the normal 64K memory available from a 16 bit address, making 1 Megabyte of memory space available. A useful feature is that PEEKing SLOT 0 will tell you your current WORKPAGE configuration.

3) MINIMAP decodes the memory into SEGMENTS so that an activating signal is available from the rear pin-wafer connections to wake up memory modules when the SEGMENT in which they are sitting is addressed.

The RAM 64 and RAM 16 modules have on-board decoding of PAGE information, so they do not have to be told in which PAGE they are sitting by patch connections, this is communicated on the Organic Bus. Both modules normally sit in the traditional ZX81 16K area, which occupies FILE A and FILE B. Should one wish to use the module exclusively in one of FILE A or FILE B then it must be told whether it is sitting in either FILE A or FILE B by patch leads from MINIMAP to the RAM module.

4) DROM, TOOLKIT and RAM 08 do not decode PAGE information appearing on the Organic Bus, and are assigned to one of the first four PAGES (PAGES 0,1,2 and 3 of BANK 0) by patch leads.

Rear pin-wafer connections

```
      TOOL        DATA       FILE A    FILE B.
    . . . . . . . . . . . . . . . . . . . .
      3 2 10     3 2 1 0
```

See memory module Technical Sheets for connection details

Using MINIMAP

MINIMAP controls which memory modules make up the WORKPAGE at any given time (see fig. 2 of "Organic Micro for the ZX81"). The WORKPAGE consists of SECTION I, which contains the ZXROM and some control areas called PATH and SLOT; SECTION II, two 16K SEGMENTS called FILE A and FILE B; and SECTION III, two 8K SEGMENTS called TOOL and DATA.

The power of MINIMAP is in being able to switch in new SECTIONS II and III from different PAGES of the memory during the execution of a program.

For example, the four 16K blocks of a RAM 64 will naturally sit in the FILE A SEGMENTS of four different memory PAGES (0 to 3). Suppose we are running a program in FILE A of PAGE 0. The system behaves exactly like a ZX81 with the 16K RAMPACK in place. However, it is possible to flip to FILE A, PAGE 1, say, during execution of a program and start running a different program there, and to subsequently flip back to PAGE 0 again, say. Since all the system variables of PAGE 0 will be unaltered by the program in PAGE 1, which has its own system variables, this corresponds exactly to a subroutine call in a high level language, and the program in PAGE 0 will continue from where it left off. It is possible to transfer data between the different programs using the TOOL and DATA SEGMENTS, since SECTIONS II and III can be flipped independently. If we had a program running in FILE A, PAGE 0, we could write data into DATA, PAGE 0. At some point during execution we could flip to PAGE 1, where a graphics program might be sitting, and use the data in DATA, PAGE 0 to generate graphical output. Then we could return to FILE A, PAGE 0 and continue our original program. Either program could write into any of four DATA SEGMENTS, a total of 32K, and make use of up to 32K of general purpose routines housed in the four TOOL SEGMENTS.

Notice that since each FILE A has its own display file flipping PAGES will instantaneously give a new display. If you flip every TV frame you will see both displays simultaneously. This means you could do fast-action graphics against a complex background just like real animators do.

So now you need to know how to:

i) Initialise a PAGE, and

ii) Flip the ingrediants of a WORKPAGE.

Page initialising

When you switch on your system it will come up in PAGE 0 of BANK 0 for all SECTIONS. This will be automatically set up by the Sinclair ROM, the system variables will be set, and a display file will be set up. The other PAGES you may have at your disposal will not have been set up at this point, and if you wish to use them later you must initialise them first. This is simply achieved by POKEing location 45440 with a number which specifies the PAGE numbers of SECTIONS II and III that will make up the new WORKPAGE. The general recipe for moving to a new WORKPAGE is 16*P2 + P3, where P2 is the PAGE number of the new SECTION II and P3 is the PAGE number of the new SECTION III. For the purposes of initialisation P3 is not relevant so set it to zero. For example, after switch on, to initialise PAGE 3 type

```
POKE 45440,48
```

There will be a short pause as the system variables are set in FILE A of PAGE 3 and the usual inverse K cursor appears, then you will be ready to type in a program. You could then initialise PAGE 2 by typeing

```
POKE 45440,32
```

and type a different program. If, after you have POKEd a new PAGE number, the screen remains "white out" indefinitely due to freak conditions in the computer, shunt the RESET on PERSONA (see Technical Sheet 1) and this will initialise the PAGE you were aiming for without affecting any of the other PAGES.

Subsequently you will want to return to PAGES you have already initialised, and to vary the SECTION III part of the WORKPAGE so that you can make use of different TOOL and DATA SEGMENTS. Just use the formula given above to instantaneously move to a new WORKPAGE configuration. For example, suppose you want a WORKPAGE which consists of FILE A from PAGE 2 and TOOL and DATA from PAGE 1. Using the formula above: 16*P2 + P3 = 16*2 + 1 = 33, so type

```
POKE 45440,33
```

and you will get the WORKPAGE you wanted. This could have been done from within a BASIC or machine code program of course.

Try the following BASIC program which demonstrates how you can move between two separately executing programs. Switch on, or move to PAGE 0, and enter:

```
10 PRINT AT 10,10;"FORGET ME"
20 PRINT AT 18,10;"/////"
30 POKE 45440,48
40 GOTO 30
```

and then type run. The program in PAGE 0 will execute and flip you into PAGE 3, where as yet there is nothing. Now enter this second program in PAGE 3

```
10 PRINT AT 10,10;"FORGET ME
NOT"
20 PRINT AT 18,10;")))))"
30 POKE 45440,0
40 GOTO 30
```

and type run. Can you see what is happening? You are running bits of program alternately in PAGES 0 and 3, and each time you flip to a new PAGE you see the display from that PAGE. You will have to type "break" to stop this figure-of-eight loop.

You can always find the current WORKPAGE configuration by typeing

```
PRINT PEEK 45440
```

Technical Sheet 4

RAM 08

Operational Description

Look at figure 1 of "Organic Micro for the ZX81". The RAM 08 module sits in either the TOOL SEGMENT or the DATA SEGMENT. The unit has all the circuitry to support up to 8K of Random Access Memory. The basic module is supplied with 2K in place, you can purchase extra 2K chips from BASICare if you wish to extend it to the complete 8K. Just plug them into the vacant sockets yourself following the simple instructions supplied.

If you decide to place your RAM 08 in a memory SEGMENT called TOOL it can be used for storing BASIC or machine code programs. You could build up a library of useful routines and load them into the normal BASIC area (FILE A) from cassette if you needed any of them, and then transfer them into the TOOL area by a series of POKES. You may have purchased a RAM 08 with the USERFONT option included. This will allow you to replace Sinclair characters with anything you like: specialist graphics shapes; lower case alphabet; foreign alphabets; or scientific symbols, for example. You could even do small high resolution pictures. Under control of the MINIMAP module up to four RAM 08 can be used in TOOL SEGMENTS (see Technical Sheet 2 on MINIMAP).

If you place RAM 08 in the DATA SEGMENT then if you have a MINIMAP you will find it most useful for transfering things between programs running in the FILE A SEGMENTS of different PAGES. This is possible because you can flip to another FILE A using MINIMAP without necessarily changing the TOOL and DATA SEGMENTS of the WORKPAGE (see figure 2 of "Organic Micro for the ZX81"). Suppose you have a program running in FILE A of PAGE 0 and you write something into DATA of PAGE 0. You can flip to FILE A of PAGE 1 whilst retaining the DATA of PAGE 0 in your WORKPAGE, and subsequently use this DATA contents as input to the second program (graphics for example).

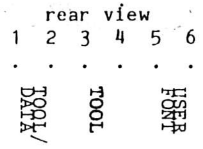
The DATA SEGMENT is immediately after the FILE A SEGMENT in memory, FILE A is where BASIC programs naturally sit. With a RAM 08 or DROM in DATA you can extend the space available for individual BASIC programs up to 24K. This is because, the variable area which comes at the end of a BASIC program can spill over into the DATA SEGMENT. So you can use very large array sizes, an array of length 1600 in just this extra space. There are two restrictions to watch out for here: i) you cannot run machine code in this area; ii) the D-FILE, the display area of a BASIC program, which is automatically set up for you by the ZXROM, must not straddle the boundary between FILE A and DATA. This is unlikely in practise, you would have to be very industrious to write that much BASIC code!

To use DATA in this way you must a) put your RAM 08 in the right place (see below) and b) let the computer know it has more headroom by changing the value of the system variable RAMTOP. If you have a 2K RAM 08 enter:

```
            POKE 16389,136
and then      NEW
```

For 4K, 6K, and 8K owners POKE values of 144, 152 and 160 respectively.

Rear pin-wafer connections



1) If you connect a shunt across pins 3&4 your RAM 08 will naturally sit in a TOOL SEGMENT. Do this if you do not yet possess a MINIMAP module.
2) If you connect a special twin-wire jumper lead from pins 1&2 of RAM 08 to the DATA selection pin on PERSONA (see PERSONA, Technical Sheet 1) your RAM 08 will be assigned to sit in a DATA SEGMENT. You should do this if you want to use the DATA SEGMENT to run a 24K BASIC program.
3) If you are a MINIMAP owner and have thus extended your options up to four TOOL SEGMENTS and four DATA SEGMENTS connect a special twin-wire jumper lead from pins 1&2 of RAM 08 to a TOOL or DATA selection pin on MINIMAP. If you only want to use one TOOL it will be easier to follow instruction 1) which will place the module in the TOOL SEGMENT of all PAGES.
4) The adventurous owners of the USERFONT option can enable it by shunting their extra pins 5&6. A program for easy character definition is supplied with the option.

For more details see Technical Sheet 6 on USERFONT.

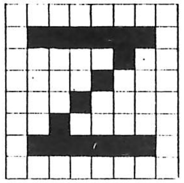
Technical Sheet 6

USERFONT

Operational Description

In the printing trade different typefaces are referred to as "fonts". BASICare now allow you to vary the characters produced by your ZX81. USERFONT is an optional extra for purchasers of RAM 08 and DROM which comes ready installed within the module. It makes it possible to define 128 of your own characters to replace the rather restricted Sinclair set of 64 characters and their inverses, you can even alter them during program execution. Now you can define special characters for graphics games, add a set of lower case characters for smarter textual output, or, by combining characters into small or large blocks do high resolution animated graphics on your screen!

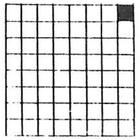
On the ZX81, the characters are made up from an 8 x 8 array of dots; and since there are 8 bits in a byte, a shape table for a character can be stored in 8 bytes. For example, look how the letter Z is constructed:



binary equivalent	hex	decimal
00000000	00	0
01111110	7E	126
00000100	04	4
00001000	08	8
00010000	10	16
00100000	20	32
01111110	7E	126
00000000	00	0

The table representing the Sinclair characters is the longest in the ZXROM, since there are 8 bytes per character and 64 characters it takes up 512 bytes of the 8K ROM. The characters are stored in the same order as in the character table of your Sinclair manual (page 181). When you enable USERFONT (see Technical Sheets 4 or 5) the ZX81 no longer looks up the character shape in the ZXROM, but in the TOOL (or DATA) SEGMENT of memory instead, in which your DROM or RAM 08 sits. The TOOL SEGMENT is the 8K block of memory immediately after the ZXROM, from 8192 decimal (2000 hex) to 16383 decimal (3FFF hex), (see Organic Micro for the ZX81, fig. 1). Now, for example when you press the letter 2, instead of the normal 2 appearing on the screen you will get whatever you may have written into the character table of your DROM or RAM 08. The letter T perhaps, or an alien invader, or part of a larger shape spread over several characters. Assuming you have assigned your DROM or RAM 08 with USERFONT option to the TOOL SEGMENT your replacement character table is actually in the second K of the TOOL SEGMENT from 9216 to 10239 decimal (2400 to 27FF hex). This means the option is available even in the minimal 2K version of the DROM or RAM 08.

As a quick example, the first eight bytes (9216 - 9223) now replace the space character, and if you POKE 9216,1 (00000001 binary) and POKE 0 into locations 9217 to 9223, if they are not already 0, your screen will be filled with tiny dots since you have replaced the blank character which normally fills the screen with the character:



(Make sure you have enabled USERFONT; that the first 2K block of DROM is not write-protected (if applicable); and that a shunt is in place on the far right of PERSONA (viewed from behind) so that the ZXROM is not "reflected" into the TOOL SEGMENT).

The USERFONT cassette

BASICare has provided you with free software to define your own characters and save them on tape. If you have a DROM, of course, it will remember for you and it may not be necessary to save them.

Load your USERFONT cassette, which has the file name "USERFONT". If you have a DROM, make sure that the first 2K block is not write-protected (shunt should be in place), as the program will modify this area. Also, make sure that the far right hand shunt on PERSONA is in place.

Do not type RUN at any time when using this program or you will erase the variables area and so lose any new characters you have created. Characters are stored by the program in an array, so when you SAVE the program after creating new characters, you also SAVE the characters. Instead, the program operates using GOTO. There are three entries into the program:

GOTO 1100

This copies the character set previously SAVEd on tape into the USERFONT area, if you have enabled USERFONT you will then be using this alternative character set. Note that it erases any characters you may have in the USERFONT area. The original program cassette contains the Sinclair character set.

GOTO 1050

This copies the Sinclair character set into USERFONT. This may be useful if you want to use some Sinclair characters augmententd with your own.

GOTO 1000

This moves the existing USERFONT contents into the array O$ in the program and then copies the Sinclair character set into USERFONT.

These routines all use FAST mode for transfering character sets and so the screen will "white out" for a few seconds.

All the options then enter the main character defining routine. This presents you with a 8 X 8 matrix of crosses on which you can define a new character. To define the position in USERFONT of the new character, you are asked to enter the character which would occupy that position if the Sinclair character set were in USERFONT. For example, if you now type "A", you will be replacing the "A" character and your new character will appear in future in place of the "A".

A cursor (inverted cross) will then appear on the matrix. You can move the cursor around the matrix using the cursor controls (keys 5, 6, 7, & 8). You can mark a particular position by pressing the "M" key when the cursor is in place, and the "D" key will delete a marked position. Positions are marked with an "O" symbol (or whatever you have replaced the "O" symbol with).

When the symbol is to your liking press "E" for End. You will now be shown both the old character and your new one, and asked which you want to retain, or whether you want to go back and alter the new one. Type "O" to keep the Old one, "N" to replace it with the New one, or "M" to Modify the new one. If you type "N" or "O" you will be asked if you want to do more symbol substitution. Answer "Y" or "N". If you type "N" you will be asked to choose whether or not to save the character set you have created. Type "N" to exit from the program, or "Y" to transfer the character set into the array O$. When the program finishes you can SAVE both the program and character set on cassette by typing SAVE in the normal way. Should you want to use the character set in the future, LOAD the cassette and type GOTO 1100.

Technical Sheet 5

DROM

Operational Description

The DROM is a very useful module indeed, because it remembers even when the main power is off. It is a memory module with 2K of memory in the cheapest version, expandable up to 8K simply by plugging in extra 2K memory I.C.'s in the vacant sockets. The extra 2K chips are available from BASICare complete with straightforward insertion instructions.

The combination of ultra low-power CMOS memory with an on-board battery to maintain the memory contents allows it to remember for over a year when the rechargeable battery is fully charged. The battery recharges itself during normal operation.

The module is useful for building up a library of BASIC or machine code routines, for storing often-used data or for temporarily storing code under development, avoiding saving and reloading from tape.

Each 2K block can be write-protected, to avoid accidentally erasing the precious contents (during a crash for example) by unplugging shunts on the rear wafer-pins.

Rear pin-wafer connections

1	2	3	4	5	6	7	8	9	10	11	12	13	14
.	.	.	.	.	.	.	.	.	.	.	.	.	.
2K no.1		2K no.2		2K no.3		2K no.4		TOOL/ DATA		TOOL		USER FONT	

1) Shunt pins 1 & 2; 3 & 4; 5 & 6; 7 & 8 to make it possible to write into the first; second; third and fourth 2K blocks respectively. Without the shunts in place the 2K blocks will be write-protected, e.g. in the case of a crash.

2) You must shunt pins 8 & 9 of the PERSONA module (see Technical sheet 1), or both the ZXROM and the DROM will both sit in the same memory space.

3) If you are using DROM with no MINIMAP then you will not have the facility of placing DROM in a particular PAGE. Shunt pins 11 & 12 to assign DROM to the TOOL SEGMENT of all PAGES. If you are using more than one module in the TOOL SEGMENT you must only enable one at a time in this way, or risk disaster.

Alternatively, attach a special twin wire jumper lead to the DATA pin on PERSONA from pins 9 & 10 of the DROM to assign DROM to the DATA SEGMENT in all PAGES. (One of the wires is an earth shield not connected at the PERSONA end).

4) If you have a MINIMAP:
and have several modules to place in TOOL SEGMENTS disconnect the shunt from pins 11 & 12 and connect a special twin wire jumper from pins 9 & 10 of DROM to a TOOL connection on MINIMAP, in whatever PAGE you want (numbered 0 to 3). If you have only one TOOL it's easier not to bother and just shunt 11 & 12. For use in the DATA SEGMENT use a special twin patch from pins 9 & 10 to a DATA connection on MINIMAP in one of PAGES 0 - 3.

5) The module can be purchased with the **USERFONT** option incorporated within the module. You can then put 128 user-defined characters into the second K (starting at 2400 hex, 9216 decimal) of the TOOL SEGMENT. When USERFONT is enabled your characters will appear in the place of the normal Sinclair characters. This is a worthwhile luxury for specialised games characters, lower case characters, foreign alphabets, scientific and mathematical characters, or high definition areas of the screen. A program for easy character definition is supplied with the option.

Enable USERFONT by shunting pins 13 & 14, if you press the keyboard now you will produce the character which is in the address position that the Sinclair character would normally occupy. E.g. the first eight bytes (2400-2407 hex) normally hold the space character. Try POKE-ing into these locations (9216-9223 decimal).

For more details see Technical Sheet 6 on USERFONT.

PERICON-a (model 1)

Introduction

The PERICON series of modules are based on the Intel 8255 A-5 chip, advanced users may wish to consult the Intel documentation. We follow Intel abbreviations in this sheet. This sheet gives enough information to use PERICON-a in most circumstances.

PERICON-a can be used for general purpose Input/Output or straightforward input. It provides three 8-bit ports (A,B & C). An 8-bit port gives you eight lines, each of which can be programmed to be in either logic state 1 or 0. Each port corresponds to a location in memory, and is set by POKEing the appropriate value, i.e. bit pattern, into the port memory location. Each line of the port could be connected to different things, e.g. LED's or relays (with buffering) or outputting data to other equipment. When used for input you simply read from a port as though it were a normal memory location by PEEKing it.

Amongst many suggestions you could use lines (when appropriately buffered) to generate sounds, or you could use some to drive seven-segment displays. Using the input and output facilities together you could drive large keypads or keyboards by scanning with PORT A and sensing with PORT B. Your train set or model car layout could be computer controlled, the positions of the trains or cars being fed back to the ZX81 using a simple sensing switch connected to an input port.

The Intel 8255 also has a control register into which you must POKE in order to specify the mode of operation of the module, i.e. which ports are input or output, and of what type.

Location in Memory

PERICON-a uses SLOT 6 (see Organic Micro for the ZX81, fig. 2). You can plug up to four PERICON-a modules onto your Organic Micro stack (devices 0-3). Each module has a base address to which you add a relative address to find that modules port and control register addresses.

BASE ADDRESS	DEVICE #
48384	0
48416	1
48448	2
48480	3

Table 1: decimal addresses of devices 0-3

RELATIVE ADDRESS	DESCRIPTION
0	PORT A
1	PORT B
2	PORT C
3	CONTROL REG (D)

Table 2: addresses of PORTS and control register relative to base addresses

The device number of a module is assigned by placing a shunt on the rear pins (see below). As an example, to write to PORT C of device 2, you should POKE into location 48448+2=48450.

The Control Word

You must write a control word into a device's control register in order to set the mode of operation of the module. There are three modes, the function of each port for each mode is summarised in Table 3:

PORT		MODE 0	MODE 1	MODE 2
	A	SIMPLE I/O	STROBED I/O	BIDIRECTIONAL
	B	"	"	N/A
	C	"	N/A	N/A

Table 3: function of each port in each mode

There are two types of control word which can be written to D, the control register:

i) $D_7=1$: When the most significant bit is set to 1 in the control word it is a mode-setting command. The rest of the word is then constructed according to Table 4.

ii) $D_7=0$: This type of control word allows direct manipulation of the lower nibble (4 bits) of PORT C. The word is POKEd into the control register and the upper nibble will be ignored. $D_0=1$ (0) will set (reset) the bit selected by a 3 bit binary number formed by D_1, D_2, D_3. Here, D_3 is the most significant bit. E.g. to set bit 3 of PORT C the lower four bits of D should be $D_3=0$, $D_2=1$, $D_1=1$, $D_0=1$.

<table>
<tr><td>D_0</td><td colspan="2">1 = input
0 = output</td><td>PORT C (lower)</td><td>NOTE 1</td></tr>
<tr><td>D_1</td><td colspan="2">1 = input
0 = output</td><td>PORT B</td><td>-</td></tr>
<tr><td>D_2</td><td colspan="2">1 = MODE 1
0 = MODE 0</td><td>MODE select for PORT B & PORT C (lower)</td><td>-</td></tr>
<tr><td>D_3</td><td colspan="2">1 = input
0 = output</td><td>PORT C (upper)</td><td>NOTE 2</td></tr>
<tr><td>D_4</td><td colspan="2">1 = input
0 = output</td><td>PORT A</td><td>NOTE 3</td></tr>
<tr><td>D_5</td><td>1=MODE 1
0=MODE 0</td><td>Dont care</td><td rowspan="2">mode selection</td><td rowspan="2">NOTE 4</td></tr>
<tr><td>D_6</td><td>0</td><td>1=MODE 2</td></tr>
<tr><td>D_7</td><td colspan="2">= 1</td><td>—</td><td>—</td></tr>
</table>

Table 4: control word construction when $D_7=1$

NOTE 1

For lower four bits of PORT C: C_0, C_1, C_2, C_3. Effective only when PORT B in MODE 0 and PORT A not in MODE 2. If PORT B in MODE 1, then C_0, C_1, C_2 are used for handshaking control. C_3 is not used by PORT B in MODE 1 & cannot be used if PORT A is in MODE 2.

In MODE 1:

C_0 - not used

C_1 - IBF (input) or $\overline{\text{OBF}}$ (output)

C_2 - $\overline{\text{STB}}$ (input) or $\overline{\text{ACK}}$ (output)

NOTE 2

For upper four bits of PORT C: C_4, C_5, C_6, C_7. Effective only when PORT A in MODE 0. If PORT A in MODE 1 input, then C_6 and C_7 can be used as defined by D_3. If PORT A is in MODE 1 output, then C_4 and C_5 can be used as defined by D_3.

In MODE 1:

C_4 - $\overline{\text{STB}}$ (input)
C_5 - IBF (input)
C_6 - $\overline{\text{ACK}}$ (output)
C_7 - $\overline{\text{OBF}}$ (output)

NOTE 3

Effective only when not in MODE 2.

NOTE 4

In MODE 2:

C_3 - not used
C_4 - $\overline{\text{STB}}$ (data into 8255)
C_5 - IBF
C_6 - $\overline{\text{ACK}}$ (data out of 8255)
C_7 - OBF

Examples:

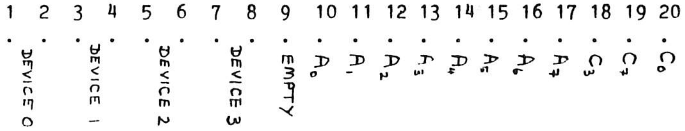
i) all MODE 0 output: D = 10000000
ii) all MODE 0 input: D = 10011011
iii) PORTS A and C (upper) MODE 0 input and PORTS B and C (lower) MODE 0 output: D = 10011000

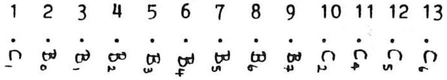
-POKEd to device base address + 3.

Reading a PORT

PORTS A, B, and C can be PEEKed. PEEKing the control register will give spurious results.

Pin-wafer connections


left hand group (viewed from rear)


right hand group (viewed from rear)

1) Assign a module to be device 0, 1, 2 or 3 by shunting one of the pin pairs 1&2, 3&4, 5&6, 7&8 respectively.

2) BASICare will supply female plugs to make connections to the rear pins.

Technical Sheet 8

PERICON-b

Introduction

The PERICON series of modules are based on the Intel 8255 A-5 chip, advanced users may wish to consult the Intel documentation. We follow Intel abbreviations in this sheet. This sheet gives enough information to use PERICON-b in most circumstances. It provides three 8-bit PORTS (A, B & C). An 8-bit PORT gives you 8 lines each of which can be programmed to be in either logic state 1 or 0. Each PORT corresponds to a location in memory, and is set by POKEing the appropriate value, i.e. bit pattern, into the port memory location.

The PERICON -b provides 24 lines of buffered output, each of which can sink 40 mA. from a 15 V. supply. This means it can be used to directly drive relays (and so control anty electrical devicee), light displays, or very long signal lines.

Location in Memory

PERICON-b uses SLOT 6 (see Organic Micro for the ZX81,' fig. 2). You can plug up to four PERICON-b modules onto your Organic Micro stack (devices 0-3). Each module has a base address to which you add a relative address to find that modules port and control register addresses.

BASE ADDRESS	DEVICE #
48256	0
48288	1
48320	2
48352	3

Table 1: decimal addresses of devices 0-3

RELATIVE ADDRESS	DESCRIPTION
0	PORT A
1	PORT B
2	PORT C
3	CONTROL REG (D)

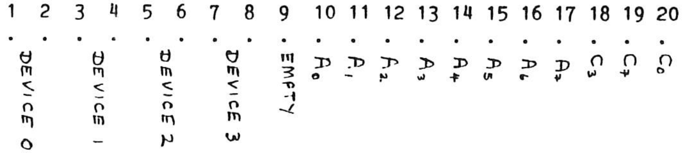
Table 2: addresses of PORTS and control register relative to base addresses

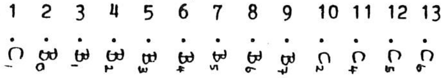
The device number of a module is assigned by placing a shunt on the rear pins (see below). As an example, to write to PORT C of device 2, you should POKE into location 48320+2=48322

The Control Word is 128

The device is set up by POKEing the control register with 128. E.g. to set up device 0 type POKE 48259,128. This must be done before writing into the PORTs.

Pin-wafer connections


left hand group (viewed from rear)


right hand group (viewed from rear)

1) Assign a module to be device 0, 1, 2 or 3 by shunting one of the pin pairs 1&2, 3&4, 5&6, 7&8 respectively.
2) BASICare will supply female plugs to make connections to the rear pins.

Organic Micro: Example Systems for the ZX81

This article demonstrates the diversity of the applications of Organic Micro. The systems described range from fundamental necessities for the hobbyist such as extra memory, to complex and sophisticated systems for business use. They represent only a fraction of the uses to which Organic Micro can be put. The construction of all systems simply involves stacking BASICare modules on top of that special module on which the rest of the system depends: the PERSONA. The PERSONA module itself connects to the ZX81 via a sturdy ribbon cable. The module is responsible for power supply and memory refresh for other modules, and can be thought of as providing basic services for the rest of the system. For more details about specific modules see the introductory sheet "Organic Micro for the ZX81" and the appropriate technical sheet on the module, which describes all aspects of its use.

You will be struck by the ease with which large systems are built up, usually requiring nothing more than adding a few modules onto PERSONA and specifying their function with small shunts which plug into pins on the rear. All system prices quoted include the cost of the PERSONA module.

Beginners' Systems

System A extra memory
Cost: £46.90
(PERSONA + RAM 16)

Plug a RAM 16 onto the PERSONA module to enlarge your memory to 16K. On PERSONA shunt pins 1&2 of the power group and pins 1&2 of the module location group. On RAM 16 shunt pins 1&2, 9&10 and 17&18.

The price may appear high for 16K of memory, but of course you are not just buying that, you will now have the capability to extend to any of the systems described below, in steps, as you can afford it. Notice also that the ribbon cable which comes with the PERSONA is of high quality, it won't disconnect in the middle of a program in the way a normal 16K add-on often does.

System B graphics fun
Cost: £51.85
(PERSONA + RAM 08 with USERFONT option)

The cheapest RAM 08 module has 2K of memory in it, the USERFONT option allows you to define your own character set of 128 characters in the second K. The first K is then free for you to store your own machine-coded programs in. You also still have the use of the 1K of RAM memory internal to the ZX81 for BASIC programs.

Now you can experiment with typefaces, create more realistic alien invaders, and you have extra memory if you want to experiment with machine code. By combining characters in blocks, and redefining them in real time, you can create hires animated graphics on your screen!

On PERSONA shunt pins 1&2 of the power group and pins 8&9 of the module location group. On RAM 08 shunt pins 3&4.

System C experimental I/O
Cost: £47.90
(PERSONA + PERICON-a)

This would be a good starter system for someone interested in experimenting with control and sensing applications, and would also be excellent for teaching purposes in schools and colleges. PERICON-a is a general purpose peripheral interface providing 24 lines of Input/Output allowing for basic I/O, strobed I/O, and bidirectional 8-bit data transfer with handshakes. You could build electronics experiments based on it, and it is easily used for e.g. sensing switches and keypads for games, or connect some LED's to it and make a logic analyser.

On PERSONA shunt pins 1&2 of the power group and pins 8&9 of the module location group. On PERICON-a shunt pins 1&2.

Specialist Systems

System D control
Cost: £52.90
(PERSONA + PERICON-b)

The 24 lies of PERICON-b are capable of directly operating relays and LED's or driving long signal lines. This system allows you to control e.g. complex light displays; model train sets; or your central heating.

On PERSONA shunt pins 1&2 of the power group and pins 8&9 of the module location group. On PERICON-b shunt pins 1&2.

Small Systems

System E computer control
Cost: £109.75
(PERSONA + RAM 16 + DROM with USERFONT + PERICON-a)

This system gives you lots of room for both BASIC and machine code programs and allows you to experiment with graphics by defining your own character sets with USERFONT. You can do Inpit/Output experiments with PERICON-a, such as control and sensing of other equipment. DROM retains its contents when the rest of the system is powered off, so you can build up a library of routines in it.

On PERSONA shunt pins 1&2 of the power group and pins 1&2 and 8&9 of the module location group. On RAM 16 shunt pins 1&2, 9&10 and 17&18. On DROM shunt pins 11&12. Shunt pins 1&2 on PERICON-a.

System F Centronics capability
Cost: £109.75
(PERSONA + RAM 16 + DROM with USERFONT + PERICON-a)

If you want to drive a professional quality 80-column dot-matrix printer which is Centronics compatible you need a PERICON-c module. This system would be useful for small code development, you can save programs you are developing in the DROM, and print them legibly. The USERFONT option will allow you to create and print specialised characters, or, by combining characters into blocks, high resolution figures. Software tools for running the printer could be permanently stored in the DROM, so you would not have not load them from cassette each session. A small business could invest in such a system for e.g. label printing.

On PERSONA shunt pins 1&2 of the power group and pins 1&2 and 8&9 of the module location group. On RAM 16 shunt pins 1&2, 9&10 and 17&18. On DROM shunt pins 11&12. Shunt pins 1&2 on PERICON-a.

System G fun and games Cost: £
(PERSONA + JOYSTICK + RAM 16 + RAM 08 with USERFONT)

With the benefit of the very superior BASICare JOYSTICK, and the USERFONT character-defining capability, your graphics games will be much more fun. You can have realistic aliens instead of black squares, and since the JOYSTICK is fully proportional, unlike others of its price, you have a heightened impression of being in control of something. The JOYSTICK is even accurate enough for digitising of graphs or figures by placing them over your TV screen.

On PERSONA shunt pins 1&2 of the power group and on the module location group shunt pins 1&2 and pins 8&9. On RAM 16 shunt pins 1&2. Shunt pins 3&4 on RAM 08.

Large Home Computer

System H serious computing
Cost: £201·20
(PERSONA + MINIMAP + RAM 64 + DROM with USERFONT + RAM 16)

This system features 88K of memory. You can have five different BASIC programs loaded at the same time and switch between them under program control. For example, the results of the simulation of an electrical network in one PAGE could be graphically presented by a different program in another PAGE. Store machine code utilities in DROM, and fancy graphics characters can be created with USERFONT (including hires screen sections). It is ideal for serious code development, even if the system crashes the DROM contents need never be erased. By switching rapidly between different PAGES of memory, each of which has its own display file, you can do multiplexed graphics e.g. complex backgrounds with animated foregrounds. Add a PERICON-c for print-out on an 80-column dot-matrix printer.

PERSONA: shunt 1&2 of the power group and pins 1&2 and 8&9 of the module location group. No shunts are needed on MINIMAP. RAM 64: shunt pins 1&2; 3&4; 5&6; 7&8; 9&10; and 11&12. DROM: Shunt pins 11&12. RAM 16: shunt pins 1&2 and 3&4 and 13&14.

Special Applications

System I Database
(PERSONA + MINIMAP + FOUR 8K DROMS + TWO RAM 64's + PERICON C)
Cost: £429·55

This system has a huge 160K of memory, as much as a floppy disc. Small business users could use such a system in database applications. For example, it could be used in stock control, mailing lists (you could do selective label printing), and accounts. A club or society could keep its records on this fast access system. A large chunk of the memory - the 32K in DROMs will stay there even when the system is powered off.

System J large-scale numerical control
Cost: £179·20
(PERSONA + DROM + 2 X PERICON-a + 2 X PERICON-b)

A powerful system suitable for numerical control of workshop machinery, lathes and machine tools. It provides 48 lines of relay driving output and 48 lines of user-defined input/output. A task can be defined via the keyboard and stored safely in DROM, where it could be easily modified. A fully developed program could be converted to EPROM and used in the TOOLKIT module.